UNITED STATES PATENT APPLICATION

for

A BLOCK QUANTIZATION METHOD FOR COLOR HALFTONING

Inventors:

Maya Rani Gupta
Michael J. Gormish
Kathrin Berkner
David G. Stork

prepared by:

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP
12400 Wilshire Boulevard
Los Angeles, CA 90025-1026
(408) 720-8598

File No.: 74451.P116

A BLOCK QUANTIZATION METHOD FOR COLOR HALFTONING

FIELD OF THE INVENTION

The present invention relates to the field of color halftoning; more particularly, the present invention relates to applying a block quantization

5   technique to color halftoning.

BACKGROUND OF THE INVENTION

Color halftoning attempts to represent an image in multi-bit (e.g., 24 bits) color as an image with only a few colors and possibly increased spatial

10   resolution. This is a problem faced by many color printers which commonly have only 5 or 8 colors available, such as, for example, cyan, magenta, yellow, black, white, and the combinations of cyan + magenta, cyan + yellow, magenta + yellow.

Judging color halftoning algorithms can be difficult due to the

15   subjective nature of vision. Specific goals in halftoning algorithms include color matching, simple computation, low graininess, artifact minimization, and reduction of post-processing effects. The relative importance of each of these goals depends upon the human viewer, the expected viewing distance, and the target image.

20   Color matching itself has at least two objectives: closeness of individual input and output colors and consistent errors. Computation

issues include running speed, parallelizability, memory size, and complexity. Graininess can be described as the ability to perceive distinct output colors (instead of the colors blending perfectly). Perceived grainess is a function, in part, of the dpi.

5   Detail information may be lost due to either poor spatial resolution or poor color resolution (or a combination of both). Artifacts are a general term for anything not desirable that is a result of the algorithm (as opposed to noise), including speckles, streaks, ghosts, gridding, texture, and very importantly, false contouring (also known as banding).

10   Post-processing effects are a concern because a halftoned image is not necessarily 'finished' when it is halftoned. Various image processing operations may still be performed on it, from downsampling to compression to edge enhancement. A halftoning algorithm may affect the success and ease of these operations.

15   Different techniques have been devised for halftoning, including dithering, error diffusion and iterative methods. Most color halftoning techniques simply extend a black-and-white halftoning technique to each of the three color planes individually. While computationally efficient, such component-wise extensions implicitly assume a mathematical linearity of

ink colors. However, the subtractive color mixing in printing is nonlinear, and thus such plane-by-plane halftoning methods are suboptimal.

Dithering consists of adding a pseudorandom 2D periodic signal to each color component of the image, then quantizing each pixel to the closest

5   output color. The dither signal is designed to generate patterns of alternating colors that blend to the eye. Different styles of dithering include clustered dot dithering, distributed dot dithering, and ordered dot dithering. Dithering with white noise does not achieve results as visually pleasing as more ordered dithering. Dithering is known to suffer from noticeable

10   periodic patterning. Many efforts have tried to improve dithering. The application of dithering to color is in the art. For more information, see A. Karun, et al., "New Methods for Dithering of Color Images," Proc. of the IEEE Workshop on Nonlinear Signal/Image Processing, 1995.

While dithering is a point process (i.e., the output at each pixel

15   depends only on the input pixel and possibly external factors like the dither signal), error diffusion diffuses the quantization error at each pixel to its neighboring pixels based on a directional spatial filter that sends a portion of the error in each forward direction. As a result, the error diffusion halftoning process adds high frequency noise. Since the goal is for the eye to

blend the colors and not notice the actual mosaic of colors, this highpass filter effect of error diffusion exploits the eyes' insensitivity to high frequency noise. Thus, error diffusion yields a blue spatial spectrum. The idea that this blue noise is more visually pleasing is well-established.

5   Error diffusion works well. However, error diffusion has some drawbacks. It is a greedy algorithm in that it locally optimizes, that is, it does the best job it can at each pixel in space. Since the error is diffused forward, it is directional and cannot provide true joint optimization over a local region. An error may get diffused far beyond the point at which the

10   eye is averaging color. Error diffusion artifacts include speckles, streaks, transient and edge effects, and grid effects at certain intensities. The feedback loop required for error diffusion can be a hindrance to parallel implementation of the algorithm.

Improvements have been made in error diffusion over many years.

15   Efforts have been made in such directions as minimizing structural artifacts, making the feedback filter adaptive, and using neural networks.

Also, there are successful black-and-white halftoning techniques that use perceptual and printer models, but these require complex extensions for color halftoning.

There are additional black and white algorithms that consider the idea of mean-matching. For example, one multiscale halftoning algorithm begins at the resolution of the whole image and sets the output image average gray level equal to the input image average gray level. Then the

5   next higher level of resolution proportionately splits up the average gray level amongst four quadrants based on their average gray levels. This process continues through the levels of resolution until each pixel is quantized black or white. For example, see Breaux, et al., "Wavelet Methods for Compression, Rendering, and Descreening in Digital Halftoning," SPIE

10   Vol. 3078, 1997; P.W. Wong, "Halftoning by Multiscale Dot Distribution," Proc. IEEE Int. Conference on Image Processing, Vol. III, 1995; P.W. Wong, "Mean matching Halftoning on a Rotated Quad-tree Structure," Human Vision and Electronic Imaging," Proc. SPIE 2657, 1996.

In another prior art solution, the average gray levels of the input and

15   output images are generated at various resolutions. In still another embodiment, wavelet averages are calculated and then dithering is used to approximate these gray-level averages. More specifically, the image is first compressed with wavelets and then the wavelet coefficients are used at each level as the average gray level to determine how white dots should be

divvied up amongst the quadrants. This technique does not go down to the pixel level and tessellates the chosen number of white dots using a clustered dot or dispersed dot dither. Their results suffer from horizontal and vertical artifacts and false contouring.

SUMMARY OF THE INVENTION

A method and apparatus for block quantization for color halftoning is described. In one embodiment, the method comprises dividing an input image and an output image into blocks, wherein each block in the output image corresponds to one block in the input image. The method also includes dividing each block into subblocks according to a criteria, calculating a color average of each input block, and calculating a set of output colors for each block in the output image to match the color average of its corresponding block in the input image.

BRIEF DESCRIPTION OF THE DRAWINGS

The present invention will be understood more fully from the detailed description given below and from the accompanying drawings of various embodiments of the invention, which, however, should not be taken to limit the invention to the specific embodiments, but are for explanation and understanding only.

Figure 1 is a block diagram of one embodiment of block quantization system for halftoning.

Figure 2 is a flow diagram of one embodiment of the process for implementing block quantization with halftoning.

Figure 3 illustrates potential output color sets.

Figure 4 is a block diagram of one embodiment of a system having block quantization processing.

Figure 5A and 5B illustrates laying out a chosen color set.

Figure 6 is an example of weighting.

DETAILED DESCRIPTION

A block quantization method and apparatus are described. In the following description, numerous details are set forth. It will be apparent, however, to one skilled in the art, that the present invention may be

5   practiced without these specific details. In other instances, well-known structures and devices are shown in block diagram form, rather than in detail, in order to avoid obscuring the present invention.

Some portions of the detailed descriptions that follow are presented in terms of algorithms and symbolic representations of operations on data

10   bits within a computer memory. These algorithmic descriptions and representations are the means used by those skilled in the data processing arts to most effectively convey the substance of their work to others skilled in the art. An algorithm is here, and generally, conceived to be a self-consistent sequence of steps leading to a desired result. The steps are those

15   requiring physical manipulations of physical quantities. Usually, though not necessarily, these quantities take the form of electrical or magnetic signals capable of being stored, transferred, combined, compared, and otherwise manipulated. It has proven convenient at times, principally for

reasons of common usage, to refer to these signals as bits, values, elements, symbols, characters, terms, numbers, or the like.

It should be borne in mind, however, that all of these and similar terms are to be associated with the appropriate physical quantities and are

5 merely convenient labels applied to these quantities. Unless specifically stated otherwise as apparent from the following discussion, it is appreciated that throughout the description, discussions utilizing terms such as "processing" or "computing" or "calculating" or "determining" or "displaying" or the like, refer to the action and processes of a computer

10 system, or similar electronic computing device, that manipulates and transforms data represented as physical (electronic) quantities within the computer system's registers and memories into other data similarly represented as physical quantities within the computer system memories or registers or other such information storage, transmission or display devices.

15 The present invention also relates to apparatus for performing the operations herein. This apparatus may be specially constructed for the required purposes, or it may comprise a general purpose computer selectively activated or reconfigured by a computer program stored in the computer. Such a computer program may be stored in a computer readable

storage medium, such as, but is not limited to, any type of disk including floppy disks, optical disks, CD-ROMs, and magnetic-optical disks, read-only memories (ROMs), random access memories (RAMs), EPROMs, EEPROMs, magnetic or optical cards, or any type of media suitable for storing electronic

5   instructions, and each coupled to a computer system bus.

The algorithms and displays presented herein are not inherently related to any particular computer or other apparatus. Various general purpose systems may be used with programs in accordance with the teachings herein, or it may prove convenient to construct more specialized

10   apparatus to perform the required method steps. The required structure for a variety of these systems will appear from the description below. In addition, the present invention is not described with reference to any particular programming language. It will be appreciated that a variety of programming languages may be used to implement the teachings of the

15   invention as described herein.

A machine-readable medium includes any mechanism for storing or transmitting information in a form readable by a machine (e.g., a computer). For example, a machine-readable medium includes read only memory ("ROM"); random access memory ("RAM"); magnetic disk storage media;

optical storage media; flash memory devices; electrical, optical, acoustical or

other form of propagated signals (e.g., carrier waves, infrared signals, digital

signals, etc.); etc.

5   **Overview of Block Color Quantization**

The present invention performs color halftoning using block

quantization with adaptively sized input blocks. Figure 1 is a block diagram

of one embodiment of a block quantization system for color halftoning. The

block quantization system performs a color match between the input

10   domain of the system and the output domain of the system to achieve color

halftoning where the number of colors in the input domain is greater than

the number of available colors in the output domain. The system performs

the color matching by dividing images in the input and output domains into

multiple blocks and matching the color average of each block in the image in

15   the input domain with a color average for a block in the image in the output

domain. The blocks may be any shape, including complex shapes that are

combinations of two or more shapes. For example, the blocks may be a

square (dxd), but they may be rectangular (dxb) where d and b are not

equal.

In one embodiment, the input domain comprises a computer system having a monitor and the output domain comprises a printer. For each small block of an input image, the block quantization technique described herein computes a set of the printing colors for a corresponding output

5   block that best renders (under a principled distance measure in one embodiment) the input block's color average. In this manner, the halftoning technique described herein ensures that for each local input region the average color matches the average color of the corresponding local output region, where the regions are small enough to ensure that the eye averages

10   over them. The block quantization technique exploits the higher spatial resolution of the printing process to achieve better color resolution and yields constrained local color optimality. Speckle, texture, and other artifacts common with error-diffusion methods are substantially avoided.

The technique treats each input color as a 3-dimensional (3D) vector

15   in color space and, in one embodiment, generates halftones using the actual output colors obtained in a subtractive printing process.

Referring to Figure 1, an input image 101 is received by a block creation and color averaging unit 102 that divides input image 101 into

blocks and generates an average color value for each of the blocks. In one embodiment, the blocks are fixed-size.

Block creation and color averaging unit 102 may include adaptive block sizing unit 103 that further divides an individual block into sub-blocks

5   if the individual block contains an edge. In one embodiment, the number of sub-blocks is four; however, any number of sub-blocks may be used. Adaptive block sizing unit 103 may iterate on the blocks, further dividing each individual block, which may be a fixed size, into sub-blocks until each block does not contain an edge or meets some other specific criteria. In

10 alternative embodiments, one, more, or all blocks may contain an edge and still not be divided into subblocks. In one embodiment, a block containing an edge may not be divided into subblocks in cases, for example, where dividing the block further causes lower color resolution. Note that such an embodiment requires feedback which has not been shown in Figure 1 to

15 avoid obscuring the present invention.

In one embodiment, to determine if each block contains an edge, the average vector color value is calculated and compared to the averages of multiple (e.g., four) sub-blocks in the block. If the difference between the block and the averages of sub-block passes a threshold, then the block is

deemed to have an edge and is further divided into the constituent

sub-blocks. This may be repeated until blocks are a predetermined size. For

example, when initially using 4x4 block pixels, the adaptive block sizing

may continue until the blocks are 1x1 blocks of pixels.

5   Given an input block's color average as a target, a color matching and

layout unit 104 computes a set of output colors that will average as close as

possible to the target value. In one embodiment, color matching and layout

unit 104 computes the output color set by examining the possible values

achievable in the color space of the output image and identifying the closest

10   achievable value to the target.

Given the output color set, color matching and layout unit 104

positioning the colors to achieve the desired color and to reduce, and maybe

even minimize, artifacts and graininess.

Figure 2 is a flow diagram of one embodiment of the processing

15   operations of the block quantization technique described herein. The

process is performed by processing logic that may comprise hardware,

software or a combination of both.

Referring to Figure 2, the process begins with processing logic

selecting a block size that the eye will average over in the input domain

(processing block 201). In one embodiment, the block size is square (dxd). In alternative embodiments, the block size is not square (e.g., a rectangle, diamond, etc.).

Next, processing logic divides the input and output images into

5   corresponding blocks (processing block 202). For each input block there is a corresponding output block with n pixels (where n depends on the output spatial resolution and the size of the input block).

Then, processing logic determines if there is an edge within each block (processing block 203). There are many ways to determine if there is

10   an edge within a block. As discussed above, in one embodiment, an edge is present if the difference between the color average of the block and its sub-blocks exceeds a threshold. If there is an edge, processing logic divides that block into sub-blocks with no edge (processing block 204). This is referred to herein as quadtreeing.

15   In one embodiment, four-by-four blocks of pixels are first considered, and then the input average of the entire 4x4 block is compared to the averages of the 2x2 sub-blocks in order to determine whether or not there is a spatial edge in the input image as part of a thresholding process. If it is deemed that there is an edge, a color match is found independently for each

2x2 block. If the thresholding process determines there is no discernible

edge, then the 4x4 block is color quantized as a whole. In one embodiment,

this process is repeated until a 1x1 block size is considered.

In one embodiment, the thresholding is based on the Euclidean

5   distance between the 4x4 average and each of the 2x2 averages. If the

Euclidean distance is over the threshold for any of the 4 sub-blocks than the

4x4 block is broken up. In general, thresholding is based on a function of the

color values in some colorspace (e.g., RGB, or some other color space).

Thresholding may compare subblock values to each other or to the whole

10   block's value.

Once the blocks no longer need to be broken into subblocks, and

assuming such an operation has not occurred earlier, processing logic

calculates the color average A_i of each input block (processing block 205). In

one embodiment, an arithmetic average over the block (equivalent to a 2D

15   Haar wavelet) is used to generate a value for each block indicative of its

color. A different wavelet or eye filter model may be used to define the

color average of blocks in the input domain, or a different sense of average

may be used, such as, for example, a geometric average or L2 average.

Furthermore, the average may be taken over the Cartesian coordinates or over another indexing of the space, such as polar coordinates.

Next, processing logic identifies a set of colors in the output domain that most closely match mathematically color averages of blocks of the input image (processing block 206). In on embodiment, processing logic computes integer coefficients β_i over the basis set of output colors B represented by the following equation of a distance measure:

$$\arg\min \text{dist}[\text{Avg}(\beta_i \cdot B), A_i] \qquad (1)$$

This operation is constrained so that each of the n output pixels is assigned a color. Thus,

$$\sum_{i=i}^{b} \beta_i = n \qquad (2)$$

where "n" is the number of pixels in the output block and "b" is the number of output colors. The same color averaging technique (equation 1 above) that is used to determine the averages of the blocks in the input image may be used as well. This process reduces, and may even minimize, a distance between the colors available in the input domain and the colors available in the output domain. Note that various distance measures may be used such as, for example, Euclidean, Manhattan, the Ll, L2, and L∞ norms.

Finally, given the set of output colors $\beta_i \cdot B$ for a given output block,

processing logic "mosaics" the chosen set of colors within the block to create

an image in the output domain (processing block 207). In one embodiment,

the processing logic attempts to best match the color of the corresponding

5   block in the input image and reduce, and maybe even minimize, spatial

artifacts. Given the set of output colors $\beta_i \cdot B$ for a given output block, the

arrangement of the colors within the block has a significant effect on spatial

artifacts and perceived color. In one embodiment, random sampling,

regular gridding, and arranging the colors using a blue noise dither matrix

10   are used, independently or adaptively switching, to achieve visually

pleasing effects. Note that an indication may be generated by processing

logic, which it sends to an output device (e.g., print engine) to dictate how

the output device is to position the chosen set of colors on the piece of paper.

The process is applicable in various color spaces, including, for

15   example, RGB and CIELAB.

The block quantization technique achieves constrained local color

optimality and does not suffer error-diffusion type artifacts. Color

matching, spatial resolution, and color resolution rival that of error

diffusion. Solving the minimization problem is the computationally

intensive part of the algorithm and may be sped up by the use of integer

programming, dynamic programming, or a look-up table. For instance, a

simple but memory intensive look-up table would take a block color value

as input and return the number of pixels to use of each of the outputs colors

5   for that block. For example, for a monitor to 4 color printer quantization, an

input block might have an average RGB value of (122, 33, 250). This could

be input to a look-up-table for blocks of size 16 output pixels. The look-up

table might return: 13 cyan, 3 magenta, 0 yellow, 0 black and 0 white. Also,

the algorithm may be sped up by finding approximate solutions, for

10   instance, by framing the problem as a matrix inversion and solving it via a

maximum entropy method or linear programming method or an

approximate look-up-table could be used. Or it could be solved

approximately by dynamic programming.

15   **A Spatial Filter and Alternatives**

An aim of certain embodiments of the block quantization techniques

described herein is to match the average color of a local input region (e.g.,

block) to the average color of the corresponding local output region. The

average of a local region can be found by applying a spatial filter over the

local region, weighting the values. An example of weighting is shown in

Figure 6. For example, a rect filter may be used. If so, the pixel in the region

is counted evenly with all the other pixels in the region.

A non-Rect filter may be used to compute the average in the input

5   domain and/or to compute the output color set.

A wavelet filter may be used to determine the average. In one

embodiment, the high-pass wavelet components may be used to determine

whether or not there was an edge. In this manner, the high-pass wavelet

filters are used for the quadtree thresholding.

10   The calculated average depends on the color space used. Different

color spaces may be used. Color spaces include, but are not limited to, RGB,

CIELAB, CIELUV, as well as Ohta, Opponent, log Opponent, HSV, HUV,

and YCbCr.

The shapes of input and output regions may be viewed with a

15   preference towards circular regions. Those shapes that strayed from a

general circular form generally provide less favorable results.

The size of the regions is selected to avoid considering regions too

large, resulting in a loss of the validity of the eye averaging assumption.

Regions too small restrict color resolution. Regions too big cause loss of

ability to resolve spatial detail. Thus, within the eye averaging size, there is

a trade-off between color resolution and spatial resolution, neglecting either

may cause loss in detail information.

5 **Choosing the Set of Output Colors**

A method to find a mathematically optimal set of n output colors

from B possible colors is described below. First, consider a simpler problem

of choosing n=3 output colors from a 2D color space delimited by B=3

possible colors: J, K, and L. Referring to Figure 3, the x's mark potential

10 choices for the set of n chosen output colors: JJJ, JJK, JKK, etc. Two vectors

JK and JL may be defined which are 1/n the length of the distance between

the "anchors" J, K, and L. The vectors JK and JL form a vector basis that

spans the 2D color space.

Gram-Schmidt orthogonalization of the vector basis may be used to

15 generate an orthogonal vector basis. A given input target color point can be

imagined as a vector from the origin J to the target. This J target vector can

be decomposed onto the orthogonal basis and reconstituted as coefficients of

the original JK and JL vectors. These two coefficients specify how many

output colors should be K and how many output colors should be L (then

the remaining colors in the set of output colors should be J). The Gram-Schmidt process results in an exact breakdown of n colors into J, K, and L to achieve the target. However, only integer solutions may be used. Therefore, in one embodiment, the closest integer solution is chosen as the

5   output color set.

With a 3D color space, the same vector technique can be used to solve for n with B equal to up to four possible colors. However, with more than four colors in 3D or more than three colors in 2D, the vector basis is redundant (e.g., the problem of decomposing the target onto the original

10   vectors exactly is undetermined, so there are many solutions although not necessarily any exact integer solutions).

For more than four colors, one solution includes considering each possible integer number of the fifth color 0 through n, decompose the (target average - fifth color contribution) onto the remaining four colors. Thus, for

15   five colors, the four color solution is found. A close integer solution is typically found unless the target color is out of the convex hull formed by the output inks.

A linear programming formulation may be used to find the appropriate set of output colors to match an input target. Linear programming

searches for a solution that reduces, or may be even minimizes, the distance

between the target color and the output color average, but the distance

reduced (minimized) cannot be the Euclidean distance (equal L2 norm

squared). Instead, linear programming reduces the Ll or sup norm.

5   Quadratic programming may be used to reduce, or may be even minimize,

the Euclidean distance.

A table look up may be used to speed up the process of choosing an

output set. This is particularly advantageous in selections where the output

colors are known beforehand, such as with, for example, printers. In one

10   embodiment, the LUT has a memory cell for every possible input color. In

an alternative embodiment, the LUT is programmed with the different

output color sets and/or an input color locates the closest output color. This

requires less memory, more processing, and may give inexact answers.

In one embodiment, the colorspace is divided into quantifiable cells

15   beforehand. All input averages that fall in a given cell are quantified to that

cell's centerpoint.

As discussed above, in one embodiment, given an input vector color

average, an output set of colors is chosen to reduce or minimize the

Euclidean distance between the input average and the chosen set of output

colors. In alternative embodiments, the Manhattan distance or

Component-Wise Max may be reduced, or may be even minimized.

As discussed above, an average is taken over the set of input colors

and the set of chosen output colors, then the Euclidean distance is reduced

5   (and may be minimized). Instead of computing an average for each block, a

distance measure (e.g., the Earth Mover's Distance (EMD)) can be used to

determine the distance between the input block and the proposed output

colors. In one embodiment, EMD is used to select between a few

possibilities that were quickly generated by some quicker output set

10   generator. For more information on the EMD, see Rubnex, Tomasi, Goibas,

ICCV 1998.

Tessellating the Output Colors

Given a chosen set of output colors, the problem remains to decide

15   how to position them down in the output space. For example, positioning 3

red pixels and 3 green pixels as

RRR
GGG

or

20   RGR

GRG

may influence the perceived color, the graininess, and the amount of false

contouring. A number of different tessellation schemes may be used,

including, for example, regular gridding, and random sampling (with and

5   without replacement), and using the values of a blue dither matrix to order

the chosen output pixels.

Regular gridding can be performed in two ways. One way is to lay

all the pixels of a given output color out in the block at once, then do the

next output color, and so on. For example, 8R and 8G would become:

10   RRRR
                                     RRRR
                                     GGGG
                                     GGGG

This method avoids isolated colors. In an alternative embodiment, a regular

15   grid may be used that alternates as much as possible. This yields well-

blended images. In this case the 8R and 8G becomes

                                     RGRG
                                     GRGR
                                     RGRG
20   GRGR

Another way to tessellate a block is random sampling. Given that 8R

and 8G are to be placed in a block, for each pixel of the output block that this

distribution may be randomly sampled, with or without replacement. This tends to lead to grainy-looking images however.

A visually pleasing way to tessellate is to lay the chosen output colors down in a blue spatial frequency pattern. One way to achieve this is to

5 choose a dither matrix designed for this purpose. The output colors are ordered in some fashion, such as, for example, by luminance. For example, the output colors could be WYCMK, where K is black. Then lay the chosen output colors down in the dither order. Figures 5A and 5B illustrate an example. If a 4x4 dither matrix looks like Figure 5A, then the chosen set of

10 output colors looks like the matrix in Figure 5B.

Dither matrices larger than a given block can be used, but then a block just uses the ordering of the values it covers.

The average may be represented better at all resolutions using either a deterministic approach or a stochastic approach that tries to more closely

15 match the averages of a block's subblocks (when the block wasn't split as discussed above with respect to thresholding) . For 4x4 blocks, after the 16 output colors are chosen, they can be divided up into four groups of four based on the average of each of the 2x2 subblocks.

Exemplary Applications

In one embodiment, the block color quantization is performed by an embedded processor in a system, such as, for example, a copier or a printer. The algorithm might be on an application specific integrated circuit (ASIC).

5 Figure 4 is a block diagram of an exemplary system that may include processing for the block quantization technique described above.

Referring to Figure 4, system 400 comprises an embedded processor 412 coupled with the embedded processor 412 for processing information. System 400 further comprises a random access memory (RAM), or other

10 dynamic storage device 404 coupled to embedded processor 412 for storing information and instructions to be executed by embedded processor 412.

Computer system 400 may further be coupled to a display device 421, such as a cathode ray tube (CRT) or liquid crystal display (LCD), coupled to embedded processor 412 for displaying information to a system user. An

15 alphanumeric input device, including alphanumeric and other keys, may also be included in the system for communicating information and command selections to embedded processor 412. An additional user input device is cursor control, such as a mouse, trackball, trackpad, stylus, or cursor direction keys, may be included in the system for communicating

direction information and command selections to processor 412, and for
controlling cursor movement on display 421.

A print engine 424 is coupled to embedded processor 412. In one
embodiment, the print engine 424 is part of a copier. Note that any or all of

5   the components of system 400 and associated hardware may be used in the
present invention. However, it can be appreciated that other configurations
of the system may include some or all of the devices.

Whereas many alterations and modifications of the present invention
will no doubt become apparent to a person of ordinary skill in the art after

10   having read the foregoing description, it is to be understood that any
particular embodiment shown and described by way of illustration is in no
way intended to be considered limiting. Therefore, references to details of
various embodiments are not intended to limit the scope of the claims which
in themselves recite only those features regarded as essential to the

15   invention.